JOBSHEET 5
Sintaks Pemilihan 2

1. Tujuan

- Mahasiswa memahami tentang operator logika
- Mahasiswa mampu menyelesaikan permasalahan dengan menggunakan sintaks pemilihan bersarang
- Mahasiswa mampu membuat sebuah program Java yang memanfaatkan sintaks pemilihan bersarang

2. Teori

Kita telah mempelajari penggunaan pernyataan IF untuk memilih sebuah tidak, pernyataan IF-ELSE untuk memilih antara dua tindakan, serta pernyataan IF-ELSE IF-ELSE dan SWITCH-CASE untuk memilih beberapa tindakan (3 atau lebih).

Terkadang kita membutuhkan pengambilan keputusan dalam bentuk level (bertingkat) sehingga di dalam suatu pernyataan IF (atau IF-ELSE) bisa saja terdapat pernyataan IF (atau IF-ELSE) yang lain. Jenis percabangan seperti ini disebut NESTED IF (percabangan bersarang).

Secara umum, bentuk penulisan pernyataan NESTED IF adalah sebagai berikut:

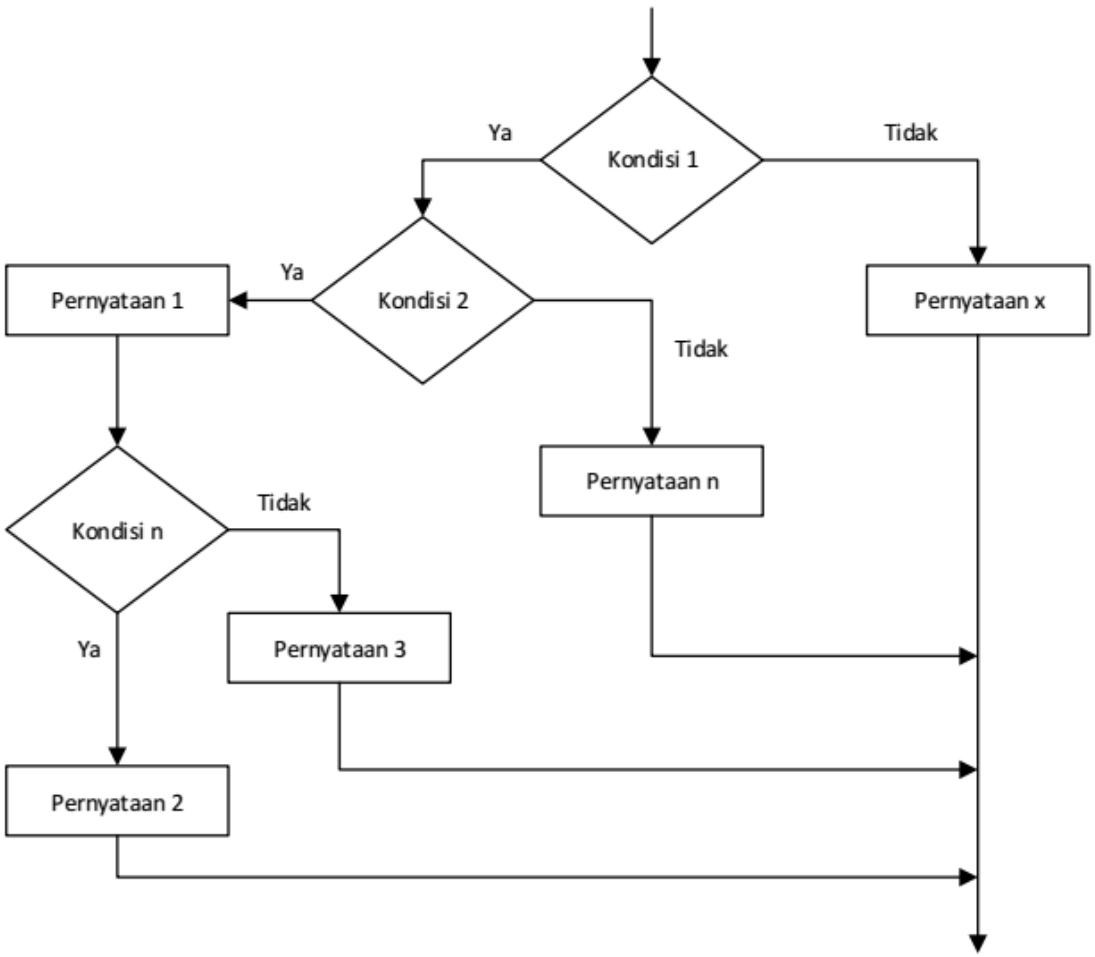
```
if (kondisi 1){
  if (kondisi 2){
    pernyataan 1;
    …
    …
    if (kondisi n){
      pernyataan 2;
    } else {
      pernyataan 3;
    }
  } else {
    pernyataan n;
  }
} else {
  pernyataan x;
}
```

Pada bentuk penulisan pernyataan NESTED-IF tersebut, kondisi yang akan diseleksi pertama kali adalah kondisi IF yang berada di posisi terluar (kondisi 1).

- Jika kondisi 1 bernilai salah, maka pernyataan ELSE terluar (pasangan dari IF yang bersangkutan) yang akan diproses. Namun, jika pernyataan ELSE (pasangan dari IF) tidak ditulis, maka penyeleksian kondisi akan dihentikan.
- Jika ternyata kondisi 1 bernilai benar, maka kondisi berikutnya yang lebih dalam (kondisi 2) akan diseleksi. Jika kondisi 2 bernilai salah, maka pernyataan ELSE (pasangan dari IF yang bersangkutan) yang akan diproses. Namun, jika pernyataan ELSE (pasangan dari IF) tidak ditulis, maka penyeleksian kondisi akan dihentikan.

Dengan cara yang sama, penyeleksian kondisi akan dilakukan sampai dengan kondisi n, jika kondisi-kondisi sebelumnya bernilai benar.

Flowchart sintaks pemilihan bersarang ditunjukkan pada Gambar 1.



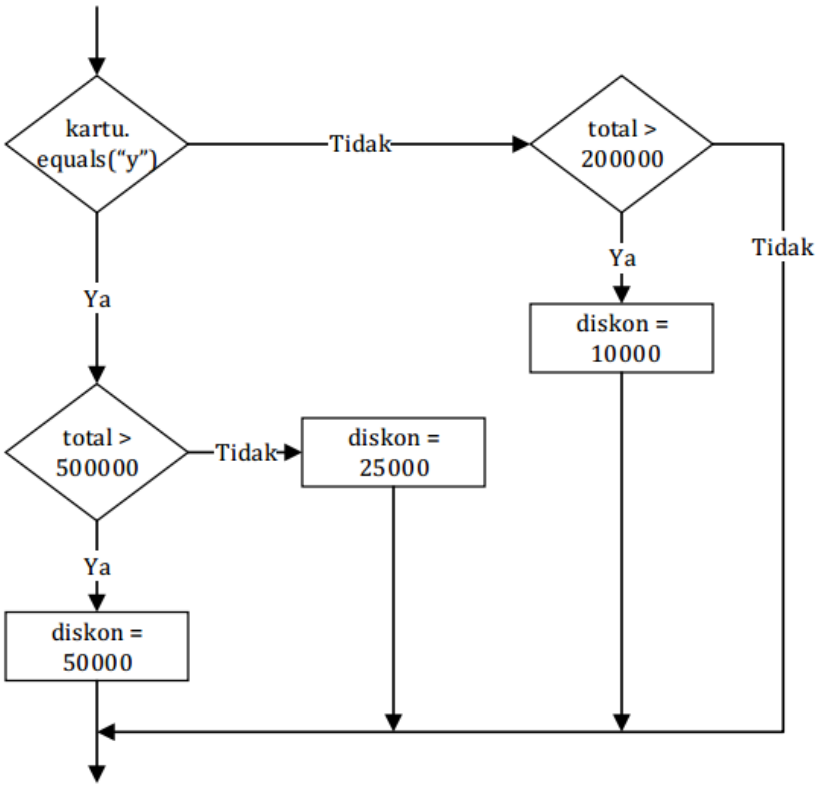
Gambar 1. Flowchart Sintaks Pemilihan Bersarang

Berikut ini adalah contoh penggunaan NESTED IF ketika seseorang akan melakukan pembayaran di kasir. Kasir akan memberikan pertanyaan sebagai berikut:

Apakah pelanggan mempunyai kartu anggota?

- TRUE: Pelanggan mempunyai kartu anggota
  - **Apakah total harga barang belanjaan lebih dari Rp 500.000?**
    - ❖ TRUE: Total harga barang belanjaan lebih dari Rp 500.000
      Pelanggan mendapatkan diskon Rp 50.000
    - ❖ FALSE: Total harga barang belanjaan tidak lebih dari Rp 500.000
      Pelanggan mendapatkan diskon Rp 25.000
- FALSE: Pelanggan tidak mempunyai kartu anggota
  - **Apakah total harga barang belanjaan lebih dari Rp 200.000?**
    - ❖ TRUE: Total harga barang belanjaan lebih dari Rp 200.000
      Pelanggan mendapatkan diskon Rp 10.000
    - ❖ FALSE: Total harga barang belanjaan tidak lebih dari Rp 200.000
      Pelanggan tidak mendapatkan diskon

Untuk lebih memperjelas alur percabangan pada contoh kasus tersebut, perhatikan flowchart pada Gambar 2.



Gambar 2. Contoh Flowchart

Gambar 3 menunjukkan kode program untuk penggunaan NESTED IF pada contoh kasus pembayaran di kasir.

```
import java.util.Scanner;
public class kasir {
    public static void main(String[] args) {
        int total, diskon, bayar;
        String kartu;
        Scanner sc = new Scanner (System.in);
        System.out.print("Apakah pelanggan mempunyai kartu anggota (y atau t)? ");
        kartu = sc.nextLine();
        System.out.print("Berapa total harga barang belanjaan? Rp ");
        total = sc.nextInt();
        if (kartu.equals("y")) {
            if (total > 500000) {
                diskon = 50000;
            } else {
                diskon = 25000;
            }
        } else {
            if (total > 200000) {
                diskon = 10000;
            } else {
                diskon = 0;
            }
        }
        bayar = total - diskon;
        System.out.println("Total yang harus dibayar: Rp " + bayar);
    }
}
```

Gambar 3. Contoh Kode Program

Pada kode program tersebut, kasir diminta untuk memasukkan input, apakah pelanggan mempunyai kartu anggota atau tidak. Selanjutnya kasir juga perlu memasukkan total harga barang belanjaan. Kondisi yang akan diseleksi pertama kali adalah nilai dari variabel "kartu". Jika pengguna memasukkan input "y", maka seleksi kondisi ini bernilai benar, dan selanjutnya dilakukan penyeleksian total harga barang belanjaan untuk menentukan diskon yang diperoleh. Gambar 4 menunjukkan hasil keluaran program ketika dijalankan.

```
run:
Apakah pelanggan mempunyai kartu anggota (y atau t)? y
Berapa total harga barang belanjaan? Rp 250000
Total yang harus dibayar: Rp 225000
BUILD SUCCESSFUL (total time: 5 seconds)

run:
Apakah pelanggan mempunyai kartu anggota (y atau t)? t
Berapa total harga barang belanjaan? Rp 300000
Total yang harus dibayar: Rp 290000
BUILD SUCCESSFUL (total time: 5 seconds)
```

Gambar 4. Contoh Hasil Keluaran Program

Kondisi di dalam pernyataan IF-ELSE dapat berupa ekspresi boolean yang kompleks, dimana operator logika seperti **&&**, **||**, dan **!** dapat digunakan. Operator yang diterapkan pada sub-ekspresi akan dievaluasi dari kiri ke kanan.

- Ketika mengevaluasi (e_1 && e_2), jika e_1 menghasilkan FALSE, maka e_2 tidak akan dievaluasi. Dengan demikian, nilai seluruh ekspresi (e_1 && e_2) akan dianggap salah. Namun, jika e_1 menghasilkan TRUE, maka selanjutnya e_2 akan dievaluasi untuk menentukan nilai seluruh ekspresi
- Ketika mengevaluasi (e_1 || e_2), jika e_1 menghasilkan TRUE, maka e_2 tidak akan dievaluasi. Dengan demikian, nilai seluruh ekspresi (e_1 || e_2) akan dianggap benar. Namun, jika e_1 menghasilkan FALSE, maka selanjutnya e_2 akan dievaluasi untuk menentukan nilai seluruh ekspresi

3. Praktikum

3.1 Percobaan 1

1. Jalankan java Buka text editor kemudian simpan dengan nama PercobaanNilai.java
2. Buatlah struktur dasar java (membuat class dan program main).
3. Tambahkan import library Scanner.
4. Buatlah struktur kondisi seperti dibawah ini :

```
import java.util.Scanner;

public class PercobaanNilai{
    public static void main (String[]args){

        int nilai;
        String grade;
        Scanner scan = new Scanner (System.in);

        System.out.print ("Inputkan nilai: ");
        nilai = scan.nextInt();

        if (nilai >=0 && nilai <=100){
            if (nilai >=90 && nilai <=100) {
                System.out.println ("Nilai A, EXCELLENT");
            }else if (nilai >= 80 && nilai <= 89){
                System.out.println ("Nilai B, Pertahankan Prestasi Anda");
            }else if (nilai >=60 && nilai <=79){
                System.out.println ("Nilai C, Tingkatkan prestasi Anda");
            }else if (nilai >=50 && nilai <=59){
                System.out.println ("Nilai D, Tingkatkan belajar Anda");
            } else {
                System.out.println ("Nilai E, Anda tidak lulus!");
            }
        }else{
                System.out.println ("Nilai yang anda masukkan tidak valid");
        }
        }
        }
```

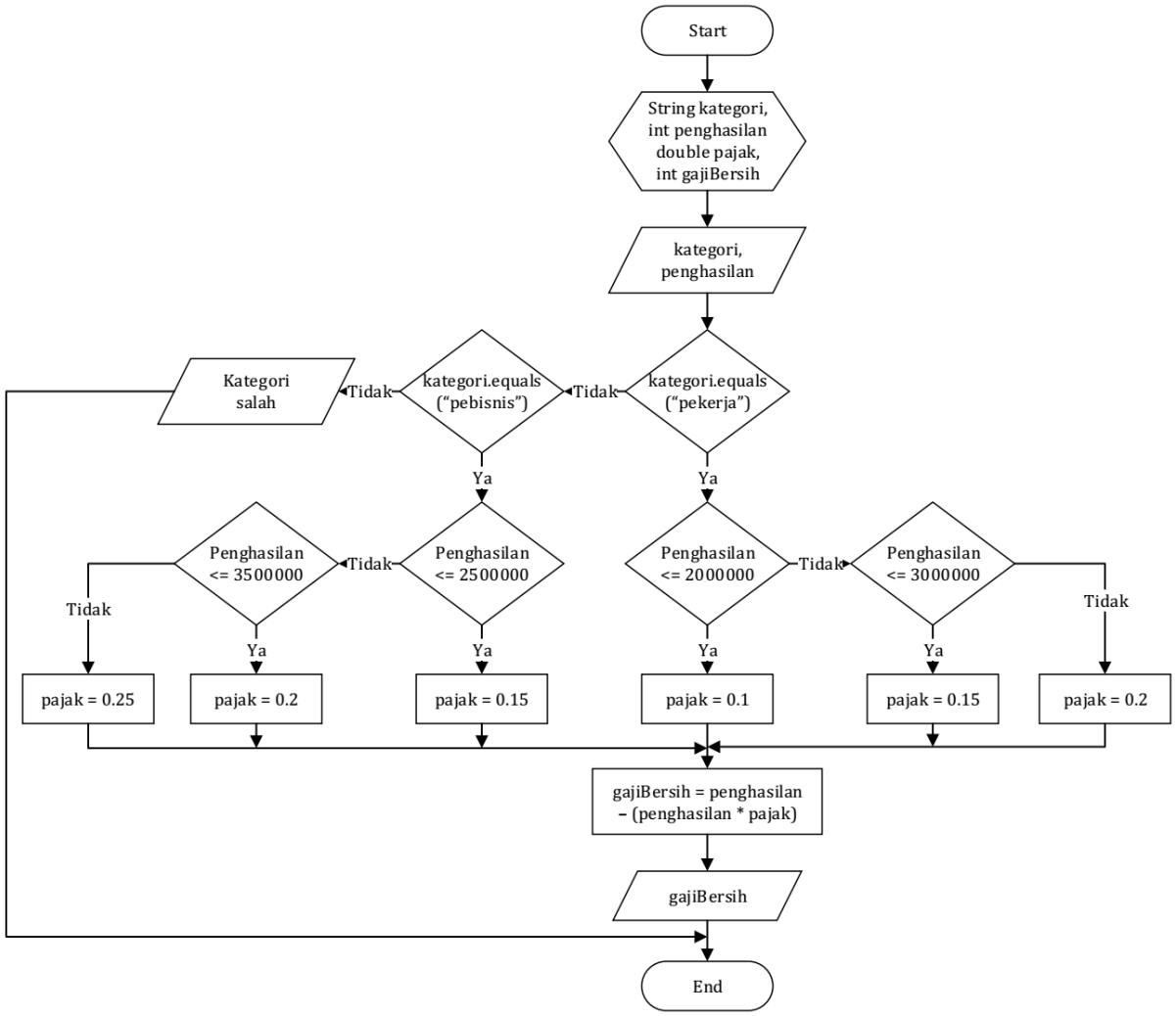
5. Jalankan program tersebut. Amati apa yang terjadi!

Pertanyaan

1. Jelaskan fungsi sintaks **`if (nilai >= 0 && nilai <= 100)`**!
2. Modifikasi kode program pada Percobaan 1 sehingga jika nilai yang dimasukkan kurang dari 0 akan ditampilkan output "Nilai yang Anda masukkan kurang dari 0" dan jika nilai yang dimasukkan lebih dari 100 akan ditampilkan output "Nilai yang Anda masukkan lebih dari 100"!
3. Ubah operator **&&** menjadi **||** pada sintaks **`if (nilai >= 0 && nilai <= 100)`**. Jalankan program dengan memasukkan nilai = 105. Amati apa yang terjadi! Mengapa hasilnya demikian?

3.2 Percobaan 2

1. Perhatikan flowchart berikut ini!



Flowchart tersebut digunakan untuk menghitung gaji bersih seseorang setelah dipotong pajak sesuai dengan kategorinya (pekerja dan pebisnis) dan besarnya penghasilan.

2. Jalankan textEditor
3. Buat nama Project **PercobaanPemilihan2.java**
4. Tambahkan import library Scanner
5. Tuliskan struktur dasar bahasa Java yang berisi fungsi main()
6. Deklarasikan Scanner dengan nama **scan**
7. Deklarasikan variabel **kategori**, **penghasilan**, **gajiBersih**, dan **pajak**;

```
String kategori;
int penghasilan, gajiBersih;
double pajak = 0;
```

8. Tambahkan kode berikut ini untuk menerima input dari keyboard

```
System.out.print ("Masukkan kategori: ");
kategori = scan.nextLine();
System.out.print ("Masukkan besarnya penghasilan: ");
penghasilan = scan.nextInt();
```

9. Buatlah struktur pengecekan kondisi bersarang. Pengecekan pertama digunakan untuk mengecek kategori (pekerja atau pebisnis). Selanjutnya dilakukan pengecekan kedua untuk menentukan besarnya pajak berdasarkan penghasilan yang telah dimasukkan. Kemudian tambahkan kode program untuk menghitung gaji bersih yang diterima setelah dipotong pajak

```
        if (kategori.equalsIgnoreCase("pekerja")){
            if (penghasilan <= 2000000){
                pajak = 0.1;
            } else if (penghasilan <= 3000000){
                pajak = 0.15;
            } else {
                pajak = 0.2;
            }
        gajiBersih = (int)(penghasilan-(penghasilan*pajak));
        System.out.println ("Gaji bersih yang anda terima: "+gajiBersih);
        }else if (kategori.equalsIgnoreCase("pebisnis")){
            if (penghasilan <= 2500000){
                pajak = 0.15;
            } else if (penghasilan <= 3500000){
                pajak = 0.2;
            } else {
                pajak = 0.25;
            }
        gajiBersih = (int)(penghasilan - (penghasilan*pajak));
        System.out.println ("Gaji bersih yang Anda terima: "+ gajiBersih);
        } else {
            System.out.println ("Kategori yang Anda masukkan salah");
        }
    }
}
```

10. Jalankan program tersebut. Amati apa yang terjadi!

Pertanyaan

1. Jalankan program dengan memasukkan kategori = pekerja dan penghasilan = 2048485. Amati apa yang terjadi! Mengapa angka di belakang koma tidak ditampilkan?
2. Jelaskan fungsi dari **`(int)`** pada sintaks `gajiBersih = (int) (penghasilan - (penghasilan * pajak));`
3. Jalankan program dengan memasukkan kategori = PEBISNIS dan penghasilan = 2000000. Amati apa yang terjadi! Apa kegunaan dari **equalsIgnoreCase**?
4. Ubah **equalsIgnoreCase** menjadi **equals**, kemudian jalankan program dengan memasukkan kategori = PEBISNIS dan penghasilan = 2000000. Amati apa yang terjadi! Mengapa hasilnya demikian? Apa kegunaan dari **equals**?
5. Modifikasi kode program pada Percobaan 2 sehingga jika penghasilan yang dimasukkan 0 atau kurang dari 0, maka terdapat informasi yang menyatakan bahwa penghasilan yang dimasukkan tidak valid!

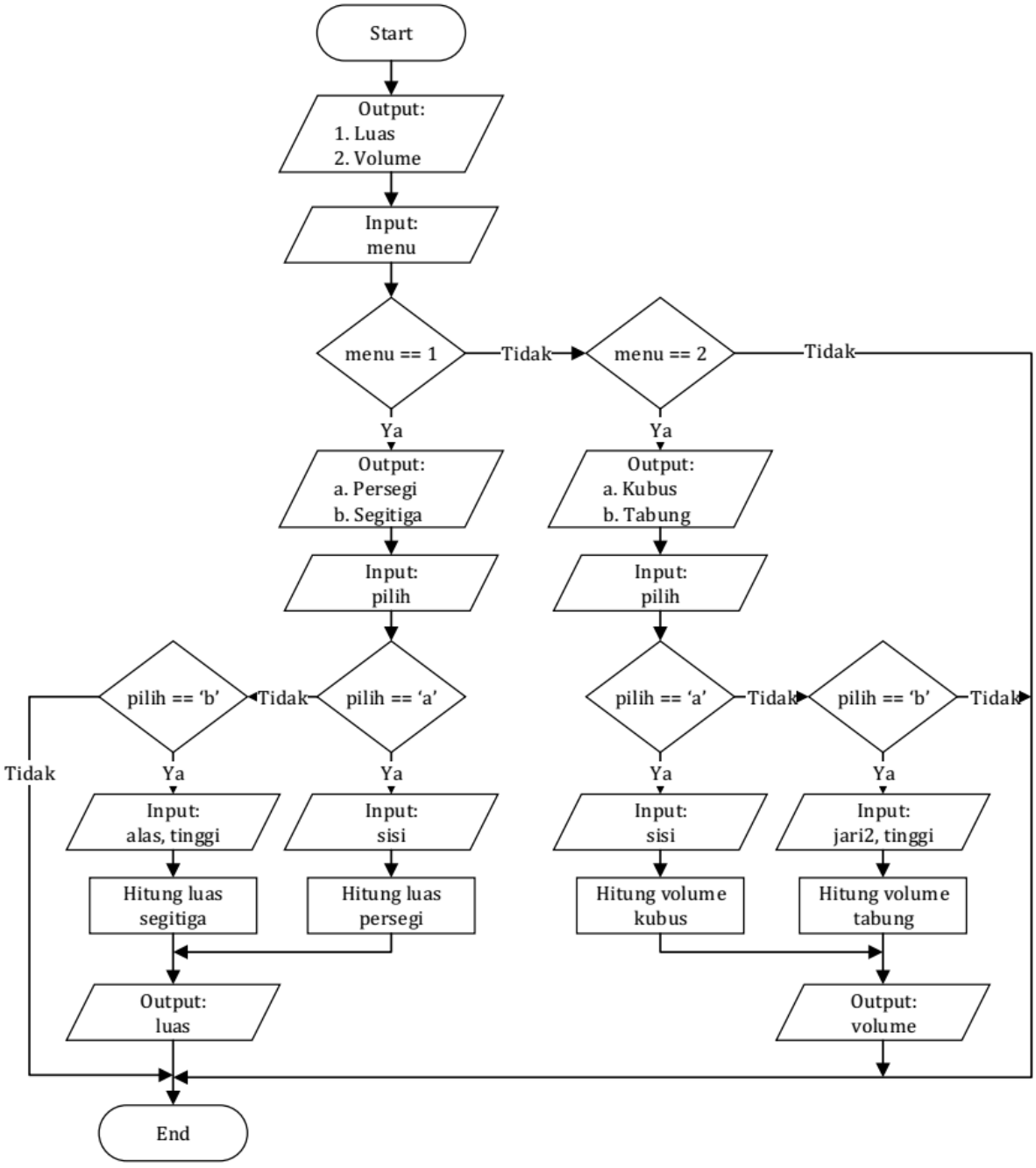
Tugas

1. Sistem pembelian tiket kereta api secara online pada kelas eksekutif Jakarta-Malang tersedia pada harga Rp 1.250.000,- pada kode keberangkatan: PG. Sedangkan pada bisnis maka biaya Rp 1.375.000 (kode keberangkatan: SG). Yang terakhir, kelas luxury terpatok pada harga Rp 2.500.000, (kode keberangkatan: MG). Pada pembelian pada setiap kelas dikenakan biaya administrasi sebesar Rp. 50.000,-. Buatlah sebuah program dimana menampilkan output :
   A. Kode keberangkatan
   B. Tipe kelas kereta yang dibeli
   C. Biaya Tiket
   D. Total Bayar beserta administrasi
   E. Selain itu masuk ke biaya regular yaitu hanya membayar Rp 50.000,-

   Contoh keluaran :

```
Masukkan Kode Keberangkatan [pg/sg/mg/reg] : mg
Kode Keberangkatan : mg
Kelas Kereta   : Luxury
Biaya Tiket    : 2500000
Total Bayar    : 2550000
```

2. Perhatikan flowchart berikut ini!



Buat program sesuai dengan flowchart tersebut!